Quantifying Information Leakage in Document Redaction

Daniel Lopresti

Computer Science & Engineering
Lehigh University
Bethlehem, PA 18015, USA
lopresti@cse.lehigh.edu

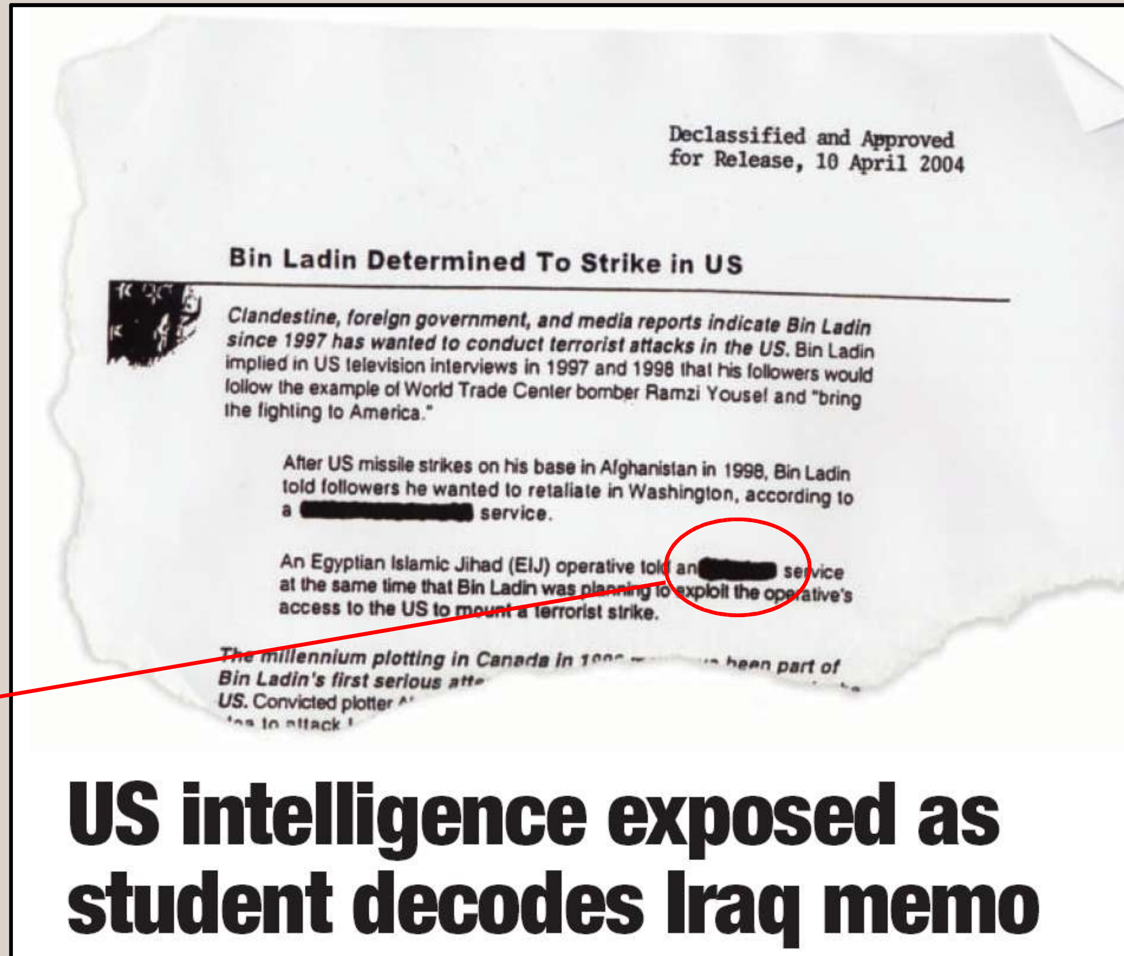
A. Lawrence Spitz

DocRec Ltd
34 Strathaven Place
Atawhai, Nelson, New Zealand
spitz@docrec.com

Motivation

Work by computer security researchers David Naccache and Claire Whelan as reported in *Nature*, May 2004.

“Egyptian” =



Declassified and Approved
for Release, 10 April 2004

Bin Ladin Determined To Strike in US

Clandestine, foreign government, and media reports indicate Bin Ladin since 1997 has wanted to conduct terrorist attacks in the US. Bin Ladin implied in US television interviews in 1997 and 1998 that his followers would follow the example of World Trade Center bomber Ramzi Yousef and "bring the fighting to America."

After US missile strikes on his base in Afghanistan in 1998, Bin Ladin told followers he wanted to retaliate in Washington, according to a [redacted] service.

An Egyptian Islamic Jihad (EIJ) operative told an [redacted] service at the same time that Bin Ladin was planning to exploit the operative's access to the US to mount a terrorist strike.

The millennium plotting in Canada in ... been part of Bin Ladin's first serious ... US. Convicted plotter ...

US intelligence exposed as student decodes Iraq memo

D. Butler, “US intelligence exposed as student decodes Iraq memo,” *Nature*, 429:116, May 2004.

Is this a topic for research?

Brings together known techniques from document analysis and natural language processing in novel, perhaps interesting ways.

Some issues:

- No access to real pre-redacted data (of course) – it's confidential. Instead, must make assumptions and hope they're reasonable.
- E.g., leaks are unintended, not strategic (but that's also interesting).
- Optimization problem – minimal redaction needed to declassify.
- Attack need not be fully automated – semi-automated is sufficient.

Ultimate goals:

- Develop understanding of how (and how much) information leaks.
- Design PASS / FAIL test for deciding if OK to release document.

How might information leak?

- Text not completely obliterated. E.g., reflective qualities of "black" may differ for laserprinter toner and marker pen.
- While obscured, certain features still deducible. E.g., numbers and locations of ascender and descender characters.
- Exploiting string set-width in monospaced fonts (e.g., Courier). Combined with language modeling techniques, this can reveal missing text or at least limit possibilities.
- Exploiting string set-width in proportionally-spaced fonts (e.g., Times). Surprisingly, this reveals even more information ...

Tools for mounting attacks

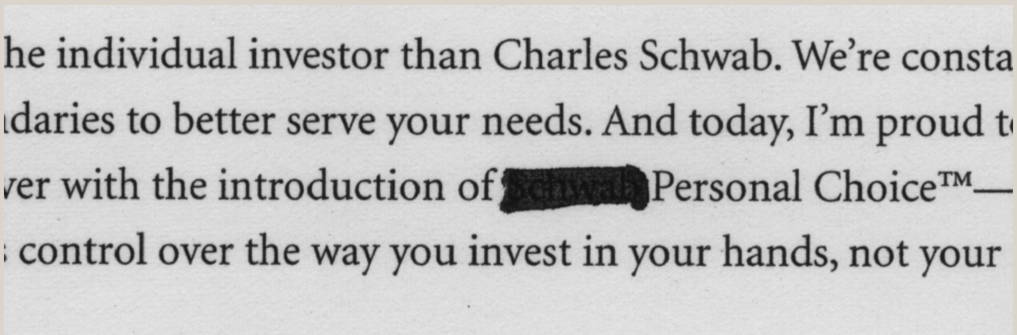
- Image processing. Apply same sorts of techniques we already use in document analysis. E.g., histograms and adaptive thresholding.
- Font metrics. Many documents prepared using one of a few, well-known fonts. Font metric data is easily available (e.g., Adobe Font Metrics files). Naccache and Whelan first did font ID via simple image processing, then applied language modeling.
- Artifacts. Knowledge of ascenders, descenders, and i-dots may be sufficient to apply Character Shape Coding (Spitz, et al.).
- Natural Language Processing. Applicable in all of above cases. Public domain tools exist for text processing (e.g., tokenization, part-of-speech tagging). Internet makes building lexicons easy.

A. L. Spitz, "Progress in document reconstruction," *International Conference on Pattern Recognition*, pp. 464-467, 2002.

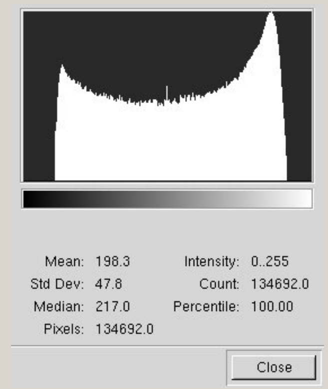
Simple image processing attack

Exploit differences in reflective qualities of "black."

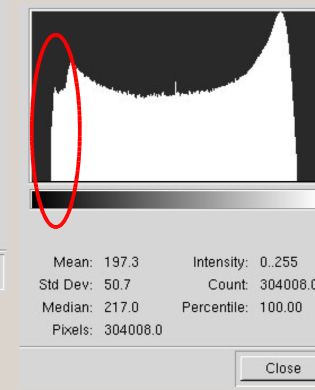
Obscure with black marker pen, then photocopy ...


he individual investor than Charles Schwab. We're consta
daries to better serve your needs. And today, I'm proud t
ver with the introduction of [redacted] Personal Choice™—
control over the way you invest in your hands, not your

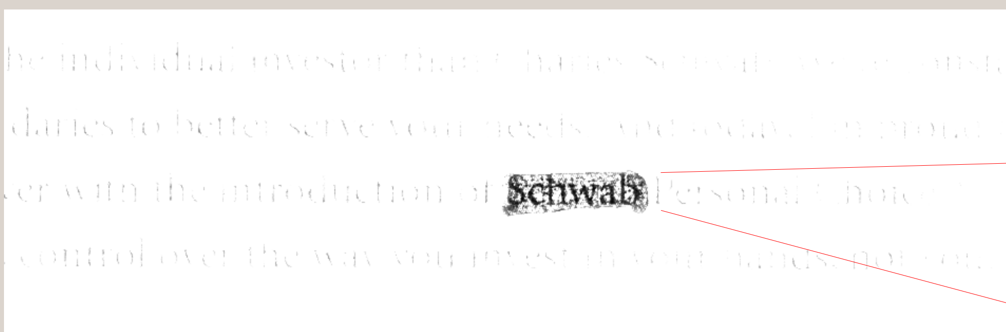
Histogram w/o redaction

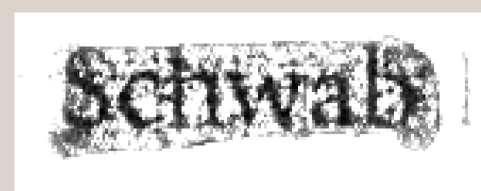


Histogram w/ redaction



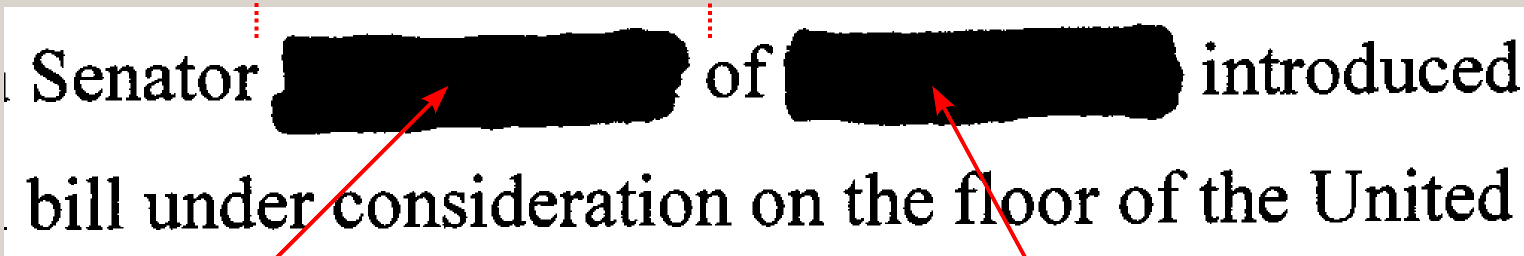
Threshold for redaction ...





What does string set-width reveal?

= width of two spaces
+ Senator's name


Senator [redacted] of [redacted] introduced
bill under consideration on the floor of the United

Likely name
of US Senator

Probably
his/her state

Preliminary experiments:

- Collect sample lexicons and font metric data.
- Study range of possible string set-widths and what it tells us.

Sample lexicons (all public domain)

YAWL

- “Yet another word list” is list of over 264,000 English words.

COUNTRIES

- 416 country names from around world (official and informal).

CONGRESS

- Names of 101 Senators (including VP) and 439 Representatives currently serving in U.S. Congress.

NAMES

- Cross-product of two lists from U.S. Census Bureau. First is list of male (1,219) and female (4,275) first names, while second is list of last names (88,799). Total of 487,861,706 names are generated.

Preliminary evaluation 1

Take Adobe Font Metrics files for Times, Helvetica, and Courier and count average number of strings of given set-width:

		Font	
Lexicon (Size)	*Times*	*Helvetica*	*Courier*
YAWL (264,057)	290	261	548
COUNTRIES (416)	1.33	1.28	1.97
CONGRESS (540)	1.66	1.60	2.90
NAMES (487,861,706)	481,125	427,573	969,903

Conclusions:

- With reliable small lexicon, attacker nearly always succeeds.
- Courier is "safer" font than Times or Helvetica (counter-intuitive).

Preliminary evaluation 2

Instead of average case, now look at worst-case analysis – number of strings which share same width with at most two other text strings:

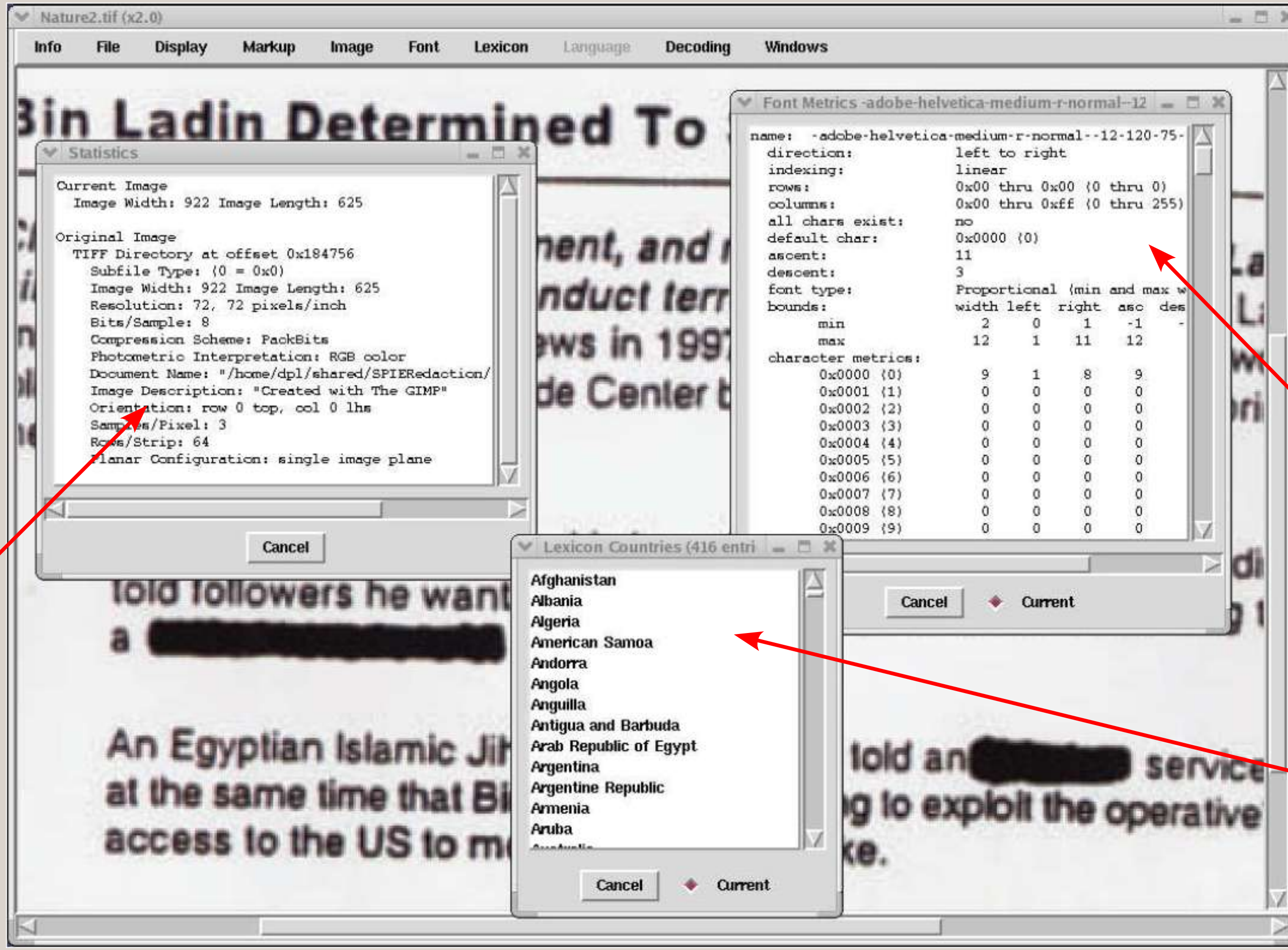
Lexicon (Size)	*Times*	*Font* *Helvetica*	*Courier*
YAWL (264,057)	213	234	112
COUNTRIES (416)	394	393	260
CONGRESS (540)	472	514	233
NAMES (487,861,706)	49	78	20

Conclusion: even for very large lexicons, some strings are easily exposed using such techniques.

Plumber: a tool for finding leaks

- As previously noted, we strongly believe that semi-automated approaches lead to the most effective attacks on redacted text.
- We have implemented a prototype system to test some of these ideas. *Plumber* is written in Tcl/Tk, a popular scripting language for building applications with rich graphical user interfaces.
- Plumber is based around an image browser. User interacts with page in question, marking it up to delineate regions of interest, designate suspected font, and choose lexicon resources.
- Plumber then proposes possible interpretations for redacted region. Can render candidate strings in indicated font to overlay of page image to confirm guesses.
- Also implements wild-card search using character shape codes.

Plumber screen snapshot

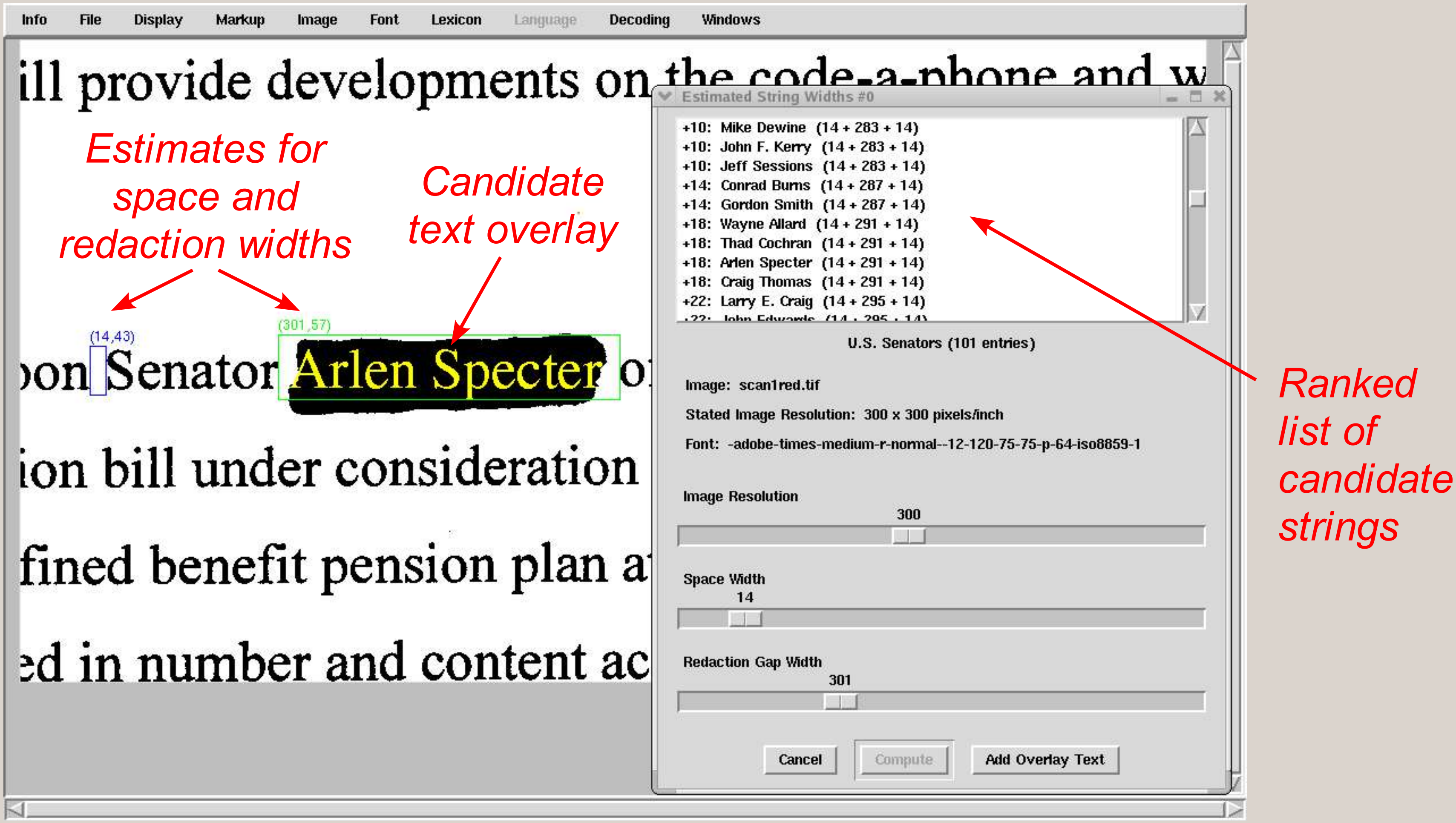


Page image statistics

Adobe Font Metrics

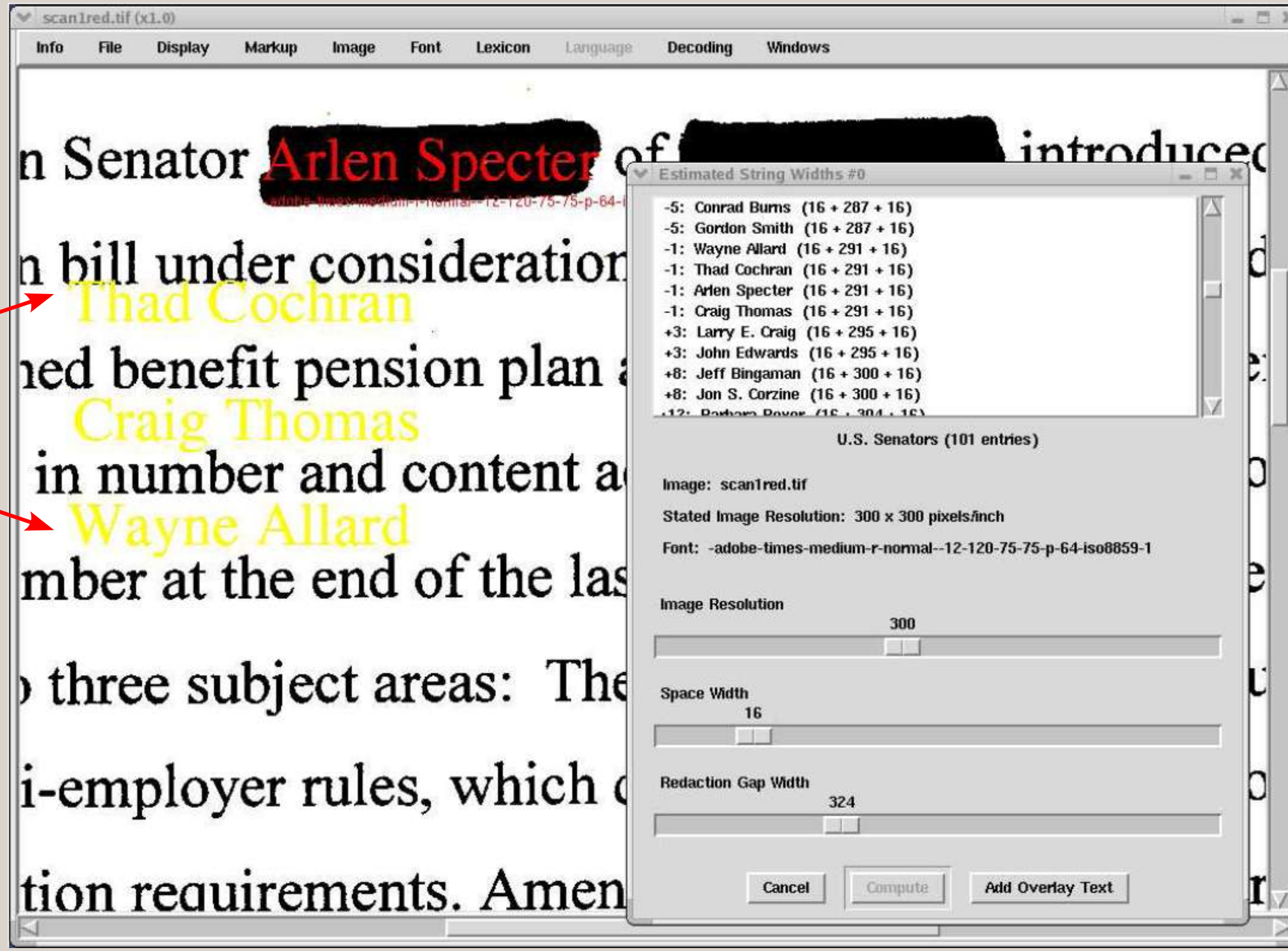
Current lexicon

Exploiting string set-width



Confirming candidate strings

Renderings of alternate candidates

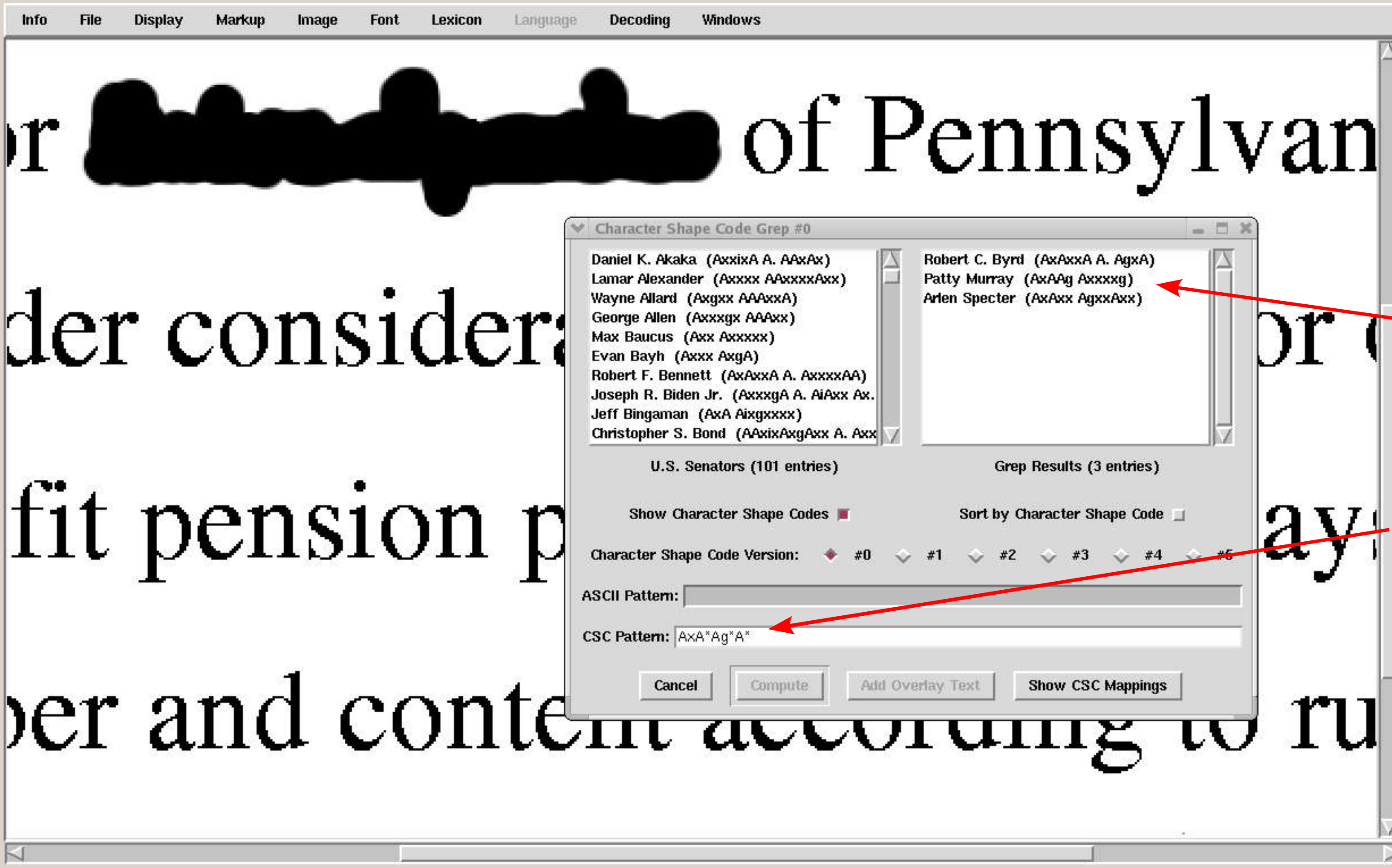


Character shape code mappings

<table>
<tr><th>Characters</th><th>V_0</th><th>V_1</th><th>V_2</th><th>V_3</th><th>V_4</th><th>V_5</th></tr>
<tr><td>amorsuvxwz</td><td rowspan="4">x</td><td rowspan="2">x</td><td>x</td><td rowspan="4">x</td><td rowspan="2">x</td><td rowspan="2">x</td></tr>
<tr><td>n</td><td>n</td></tr>
<tr><td>c</td><td colspan="2" rowspan="2">e</td><td colspan="2">c</td></tr>
<tr><td>e</td><td colspan="2">e</td></tr>
<tr><td>ACGIOQSTUVWXYZflt</td><td colspan="3" rowspan="7">A</td><td>A</td><td rowspan="7">A</td><td>A</td></tr>
<tr><td>HMN</td><td>N</td><td>N</td></tr>
<tr><td>bhkL</td><td>b</td><td>b</td></tr>
<tr><td>BDEKR</td><td>E</td><td>E</td></tr>
<tr><td>PF</td><td>P</td><td>P</td></tr>
<tr><td>dJ</td><td>d</td><td>d</td></tr>
<tr><td>gpqy</td><td colspan="6">g</td></tr>
<tr><td>i</td><td colspan="6">i</td></tr>
<tr><td>j</td><td colspan="6">j</td></tr>
</table>

A. L. Spitz, "Using character shape codes for word spotting in document images," *Shape, Structure and Pattern Recognition*, pp. 382-389, 1995.

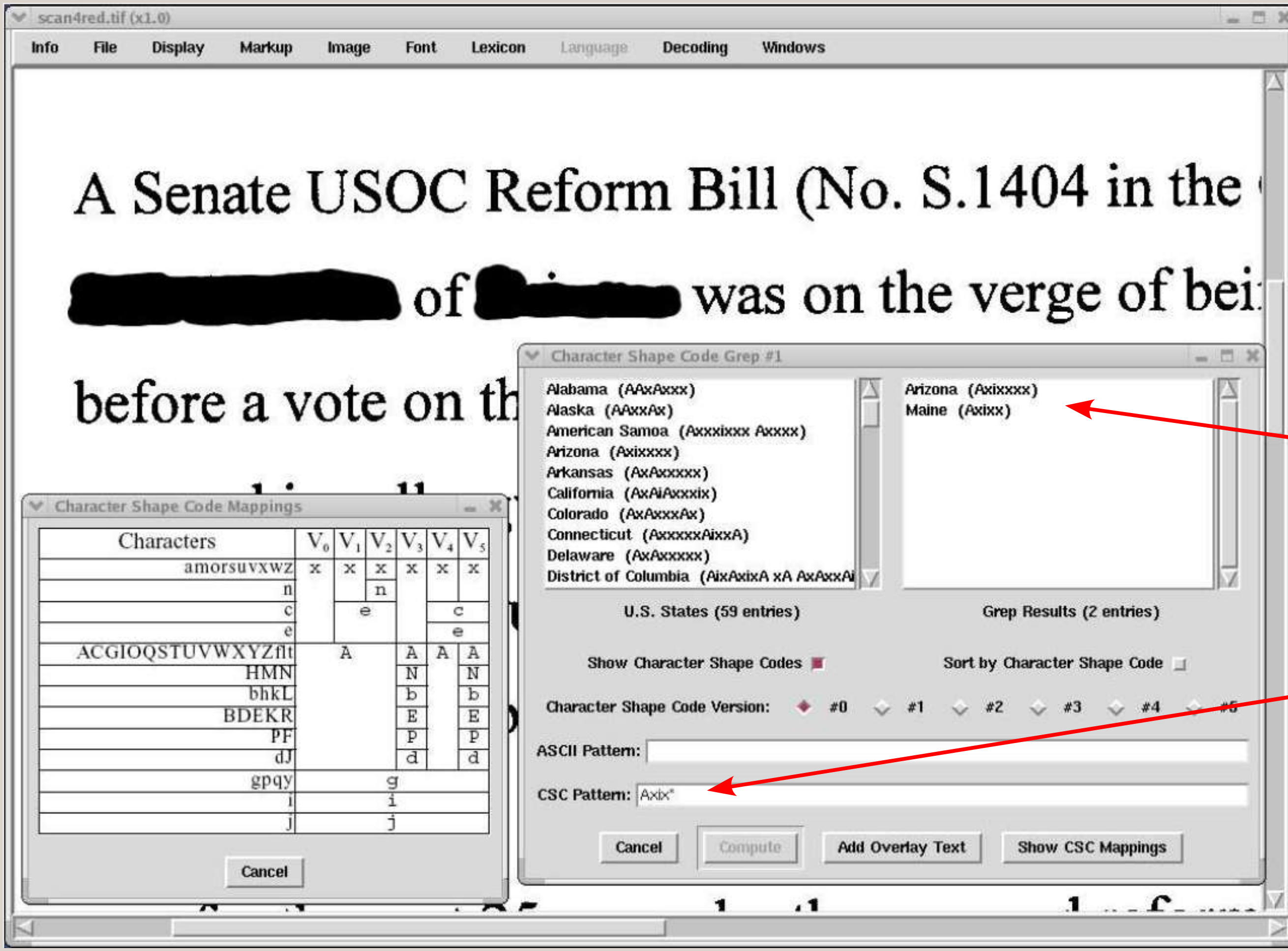
Plumber CSC wild-card search



Matching candidates

*CSC pattern = AxA*Ag*A**

Plumber CSC wild-card search



Only 2 states match: Arizona and Maine

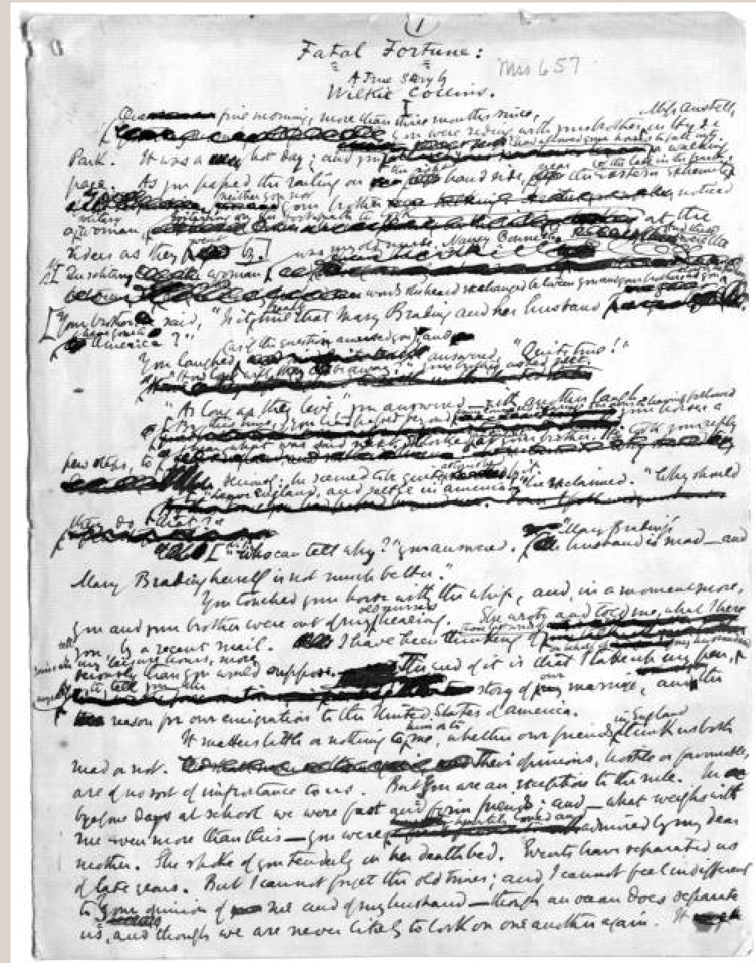
*CSC pattern = Axix**

Related (but even harder) problem

Interestingly, this problem does not appear to be unique to redacted text.

Title page from "Fatal Fortune: A True Story" by Wilkie Collins.

> "Collins' manuscript is a short story told from the perspective of a woman who falls in love with a man suspected of being 'mad.' He has been subsequently disenfranchised of his fortune by scheming executors."


Fatal Fortune:
A True Story by
Wilkie Collins.
MS 657

From "I remain: A Digital Archive of Letters, Manuscripts, and Ephemera" at Lehigh University.

http://digital.lib.lehigh.edu/remain/

Summary

Whether the problem of information leakage through redaction is of practical importance is unclear to us (and might forever remain so).

Still, there seem to be many interesting technical issues here:

- How can we quantify the amount of information leakage?
- When is a document safe for release?
- Can the process of redaction (or confirming that a document has been safely redacted) be automated?
- Is it possible to design effective counter-measures to make documents safer? E.g., special fonts or typesetting conventions that defeat set-width attacks?
- Does what we learn from studying this problem apply elsewhere?